Andrew Errington on Fri, 08 Nov 2002 08:43:40 +1300

[Date Prev] [Date Next] [Thread Prev] [Thread Next] [Date Index] [Thread Index]

Re: Not an awk question

- *To*: linux-users@HIDDEN
- *Subject*: Re: Not an awk question
- *From*: Andrew Errington <a.errington@HIDDEN>
- *Date*: Thu, 07 Nov 2002 20:44:15 +0000 (GMT)
- *Comments*: University of Canterbury Linux Users Group
- *In-reply-to*: <1036751709.13848.23.camel@zane.localdomain>
- *List-subscribe*: <mailto:mailserv@it.canterbury.ac.nz?body=subscribe%20linux-users>
- *List-unsubscribe*: <mailto:mailserv@it.canterbury.ac.nz?body=unsubscribe%20linux-users>
- *Reply-to*: linux-users@HIDDEN

```
>
> There is an entire development methodology (whose name escapes me at the
> moment) that makes use of that very phenomenon.

We called it the Rubber Duck method of debugging.  It goes like this:

1) Beg, borrow, steal, buy, fabricate or otherwise obtain a rubber duck
   (bathtub variety)
2) Place rubber duck on desk and inform it you are just going to go over
   some code with it, if that's all right.
3) Explain to the duck what you code is supposed to do, and then go into
   detail and explain things line by line
4) At some point you will tell the duck what you are doing next and then
   realise that that is not in fact what you are actually doing.  The duck
   will sit there serenely, happy in the knowledge that it has helped you
   on your way.

Works every time.  Actually, if you don't have a rubber duck you could at
a pinch ask a fellow programmer or engineer to sit in.

Andy
```

- **References**:
  - **Re: Not an awk question**
    - *From:* Zane Gilmore

- Prev by Date: **Re: MS networks (was Re: Partitioning)**
- Next by Date: **Re: thursday 14h november meeting...**
- Previous by thread: **Re: Not an awk question**
- Next by thread: **Re: Not an awk question**
- Index(es):
  - **Date**
  - **Thread**